ARMY RESEARCH LABORATORY

Allocating Tactical High-Performance Computer (HPC) Resources to Offloaded Computation in Battlefield Scenarios

by Tamim I. Sookoor, David L. Bruno, and Dale R. Shires

ARL-TR-6757 December 2013

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory

Aberdeen Proving Ground, MD 21005-5066

ARL-TR-6757 **December 2013**

Allocating Tactical High-Performance Computer (HPC) Resources to Offloaded Computation in Battlefield Scenarios

Tamim Sookoor, David Bruno, and Dale Shires

Computation and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
December 2013	Final	December 2013

4. TITLE AND SUBTITLE
Allocating Tactical High-Performance Computer (HPC) Resources to Offloaded Computation in Battlefield Scenarios

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Tamim I. Sookoor
David L. Bruno
Dale R. Shires

5d. PROJECT NUMBER
R.0006163.13

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: RDRL-CIH-S
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-6757

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

We envision a battlefield of the future where computation plays a pivotal role in providing the Army an advantage over its enemy. The ability for computationally intensive tasks to be carried out in real-time at the front lines will give Soldiers access to a wealth of information that they currently do not possess. This information could range from suggesting optimal locations from which to observe points of interest to locating snipers based on the shock waves of bullets. In order for these, and yet unknown applications, to be implemented a considerable computational capability has to be mobilized with an army into the battlefield. Mobile devices will soon become an integral part of a Soldier's toolkit as the military is attempting to introduce smartphones into combat. In order for the potential of these tools to be realized, their fundamental weakness of limited battery life has to be addressed. One approach to increasing energy efficiency of mobile devices is to offload computationally intensive applications to more capable machines. While a number of approaches to offloading have been proposed, none so far have addressed the issue of intelligently scheduling the offloaded applications to a cluster of heterogeneous machines. This paper describes the state-of-the-art in techniques necessary to implement such a scheduler and presents a plan for a project to implement such a scheduler.

15. SUBJECT TERMS
computation offloading, task allocation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Tamim I. Sookoor
Unclassified	Unclassified	Unclassified	UU	28	19b. TELEPHONE NUMBER (*Include area code*) 410-278-6993

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

1. Introduction

There has been a recent effort in the military to equip Soldiers with smartphones(*1–11*). The availability of these resources on the tactical edge enables Soldiers to be empowered through a large number of applications. These applications range from simply providing a Soldier with his or her location through the Global Positioning System (GPS) functionality of the phone, to using the built-in microphones to locate the source of a gunshot, or provide Soldiers information on the best locations from which to observe an enemy. As with any battery powered device, a major constraint when providing the Soldier with sophisticated applications on a mobile device is the limited lifetime of the device between charges.

In order for battlefield computation on mobile devices to become a reality, a number of constraints have to be overcome. The first is the environment in which it will be deployed, which could be harsh and unpredictable. The theater of operation could vary from urban terrain to deserts and the computing framework should adapt to the challenges of each environment. Mobility poses a number of challenges as it will lead to frequent changes in network topology requiring the system to be robust to intermittent connectivity. This mobile device computing framework would be used by Soldiers who may not have the technical knowledge to operate a complex system; therefore, the system should be automated as much as possible providing the Soldiers with an intuitive user interface. In order to allow a large number of applications to be developed, some of which may not even be envisioned yet, the computation framework should make application development easy by not overburdening programmers with having to annotate code or learn new programming abstractions. Finally, security is a major concern for a military system; therefore, the battlefield computing network should be hardened against malicious attacks and vulnerabilities.

One approach to maximizing the battery life of mobile devices is offloading computationally intensive applications to personal computer (PC) or server class devices. In addition to reducing the computational load on the devices, offloaded applications benefit from being able to execute faster on more capable machines and producing better output due to the availability of more resources on a PC or server. A number of projects have demonstrated the benefits of such an approach. For instance, Kemp et al. (*12*) demonstate the benefit of offloading for image, audio, and text processing as well as Artificial Intelligence (AI), 3-D rendering, and security.

Enabling offloading in a tactical situation, such as a battlefield, results in a number of challenges.

The first is providing server class devices that are accessible to the mobile devices. Offloading solutions such as Cuckoo(*12*), MAUI(*13*), COMET(*14*), and ThinkAir(*15*) offload applications via Wi-Fi or 3G networks to servers or commercial clouds such as Amazon EC2. Soldiers in a battlefield cannot rely on Wi-Fi connections or cellular networks to connect to servers or clouds. Instead, we propose utilizing mobile High-Performance Computer (HPC) resources made available at the front lines in the form of tactical cloudlets (*16*). Offloading applications from mobile devices in a battlefield to tactical HPC assets poses a number of challenges not faced by traditional offloading frameworks.

While there are aspects in these systems that are applicable to a military heterogeneous network of mobile devices and tactical HPCs, there are still a number of areas that are not sufficiently addressed. The first is the generalizability of their application development processes. Most of the existing systems have been tested on a small set of applications and the ease of use of their programming models for military applications has not been assessed. The second is the methods used for application decomposition during offloading. Different systems decompose applications at different granularities and thus, may be more appropriate for different classes of applications. A computation platform composed of mobile devices should be optimized to efficiently decompose applications that will be of most interest to the military, while being flexible enough to decompose yet unforseen applications. Existing systems rely on Wi-Fi and cellular networks for communication, which may not be available in the environments where such a system would operate. Thus, alternative means for communication are necessary. Security is not a major concern in most existing systems with reputation based approaches considered sufficient. Yet, for military applications where active cyber-attacks are a possibility, security is an important consideration. Finally, the applications targeted by most existing offloading solutions have limited requirements for reliability and robustness, while military applications have to guarantee Quality-of-Service (QoS) requirements, such as deadlines for computations to complete and system lifetimes.

2. Statement of the Scientific Problem

Offloading applications in a tactical environment is complicated by unreliable wireless communication links—so that a phone that was connected to an HPC resource at one time may not be connected at another; movement of the mobile devices and tactical HPCs resulting in the closest HPC asset changing; and varying QoS requirements for applications—such as a deadline by which an application has to complete or an energy budget that has to be met. The resource

allocation issue is further complicated by the fact that resource allocation decisions have to be made in a distributed manner by the mobile device attempting to offload an application. In order for such an allocation to be made efficiently, nodes have to be aware of the global state of the network and the HPC assets. In order to minimize the impact on the battery life of the mobile device, this global view has to be disseminated and updated with a minimum number of transmissions.

3. Background and Related Work

Some areas of research related to the resource allocation problem for computation offloading include task scheduling for heterogeneous computer systems, task allocation for mobile robots, and resource allocation in wireless sensor networks. Briceno et al. (*17, 18*) describe techniques to schedule tasks arriving in a heterogeneous computing system while ensuring the robustness of the schedule. The algorithms are robust to unknown task arrival times and can satisfy constraints such as ensuring high-priority tasks are executed before other tasks, yet requires a central task allocator that is aware of all the tasks and the available resources at any point in time. Task allocation in multirobot systems (*19, 20*) addresses the complexity associated with mobility, yet do not have the constraint of a limited battery life that is faced by a resource allocator on a mobile device. Mainland et al. (*21*) present an approach to allocate resources in a wireless sensor network in a decentralized manner. In the event of no connectivity to HPC assets, we propose using a scheme similar to that presented by Arslan et al. (*22*) or Shi et al. (*23*) to offload computation to other mobile devices. This section describes some of this work and how they relate to the task allocation problem we are addressing. We also point out why the existing solutions are insufficient for our needs.

3.1 Centralized Resource Allocation

Researchers have proposed a number of methods to allocate resources to processes, such as assigning processors to tasks or people to jobs, where a single entity, such as a server, is responsible for scheduling the resources. Such approaches are mainly used in computing clusters where all the tasks can be sent to a single machine, which knows the status of all the nodes in the cluster, and can schedule the tasks based on a scheduling algorithm. In this section we summarize some approaches for such centralized resource allocation.

Young et al. (*24*) describe an algorithm to schedule tasks on a heterogeneous cluster of machines

in order to maximize the number of task deadlines met without violating a system energy constraint. The challenges addressed by the algorithms include uncertain task execution times and the system being oversubscribed at certain times. The authors present a validated model of robustness for the computing environment and adapt two existing heuristics to utilize robustness and assign tasks without violating the energy constraints. In addition, the authors present a new heuristic for task assignments and evaluate the three heuristics via simulation. The heterogeniety of the cluster is due to differing numbers of processors per node, differing numbers of cores per processor, the available processor frequencies, power consumption profiles, and power supply efficiencies. The tasks are modeled as a dynamically arriving collection of independent tasks with the mix of tasks unknown prior to arrival. The authors assume the existance of a probability mass function (pmf) of execution times for tasks and state that such a pmf can be derived from histograms of experiments, or past runs of tasks, or obtained using analytical techniques. The sum of the random variable represented by a task's pmf and its ready time, the time at which a core can start executing it, is used as an estimation of the task completion time. This estimate is used to calculate the robustness as the number of tasks expected to complete by their deadlines. The two adapted heuristics are Shortest Queue (SQ) and Minimum Expected Completion Time (MECT). The authors define a Lightest Load (LL) heuristic and a Random (RAND) heuristic as a baseline. In an evalution involving the scheduling of a generalized filter mechanism, each heuristic improved by 10% due to the robustness metric.

Briceno et al. (*17*) present a static load balancing scheme for the satellite data processing portion of a space-based weather monitoring system. The load balancer has to allocate resources on a heterogeneous distributed processing platform to two categories of tasks as they arrive. High-Priority Tasks (HPTs) have to be allocated resources before Revenue Generation Tasks (RGTs), yet sufficient RGTs have to be executed in order to minimize the time to reach a profit. The problem is complicated by the fact that the arrival times of new data sets to be processed is uncertain and the current data has to be processed before the next set arrives. Due to these constraints, all tasks cannot be completed and the algorithm attempts to minimize the makespan of all HPTs as well as the time to break even in terms of revenue.

Kuhn(*25*) presents a solution for the "assignment problem" using the work of two Hungarian mathematicians: D. Konig and E. Egervary. The "assignment problem" assigns people to jobs so that the sum of ratings, indicating the compatibility of a person and a job for instance, is maximized. Thus, it attempts to find the best assignment of people to jobs. The author first simplifies the problem to use only two ratings, 0 and 1, indicating whether or not a worker is qualified for a job. He derives an algorithm from the proof of Konig to solve this problem. Then he shows that the general problem of assignment can be reduced to this special case with a

computationally trivial procedure derived from the work of Egervary. The Simple Assignment Problem, tackled by the author first, attempts to find the largest number of jobs that can be assigned to qualified individuals without an individual being assigned more than one job. The author views this problem from the point of view of a budget to account for the value of an individual assigned to a job for which s/he is qualified. The budget assigns either 0 or 1 to each individual and each job and the algorithm attempts to find an adequate budget where for every individual qualified for a job, the individual, the job, or both, are assigned 1. The paper describes the algorithm for finding an adequate budget and then transforming the general assignment problem to the simple assignment problem efficiently, but no performance evaluation of the algorithm is carried out.

There are a number of commercial cloud services such as Elastic Compute Cloud (EC2), which is part of Amazon Web Services (AWS) and Windows Azure. In the open-source domain, OpenStack(*26*) is being widely adopted as an Infrastructure as a Service (IaaS) for cloud computing. The project launched by Rackspace Hosting and National Aeronautics and Space Administration (NASA) aims to enable organizations to offer cloud-computing services running on standard hardware. Similar to the research projects described above, these commercial and open-source solutions also target domains very different from the tactical edge that motivates our approach. The challenges and constraints faced in an environment such as a battlefield will not be adequately served by systems that rely on stable hardware with reliable power and no mobility.

While some of the ideas presented in these papers and projects may be applicable in scheduling applications once they are queued on an HPC asset, they do not consider some of the issues that arise in a tactical cluster of mobile devices and Tactical High-Performance Computers (T-HPCs). Our system has much greater heterogeneity than a cluster of servers with different configurations of processors. Also, these approaches do not consider unreliable networks due to the servers communicating over reliable wired links. Mobile devices in the battlefield are expected to have unreliable communication and communication unreliability increases with complexity as tactical cloudlets are introduced into the system. The centralized scheduler that these methods require is not feasible in a tactical scenario and the mobility of our application is not an issue in these approches.

3.2 Distributed Resource Allocation

Lim et al. (*27*) tackle the problem of allocating the limited sensing, processing, and communication resources in a wireless sensor network, without a central coordinator, in order to minimize costs and maximize network capability. The solution, Adaptive Distributed Resource

Allocation (ADRA), uses simple local actions performed by individual nodes for mode management. This enables each node to adapt its operation over time in response to the status and feedback of neighbors giving rise to the desired global behavior. Distributed real-time resource allocation is complicated due to the following four reasons: (i) a large number of decision makers, (ii) limited communication among decision makers, (iii) dynamically changing environment, and (iv) a time constraint on the solution. ADRA is a heuristic to guide sensor network nodes to efficiently allocate resources.

Mainland et al. (*21*) present Self-Organizing Resource Allocation (SORA), a novel algorithm to allocate limited resources on sensor nodes in order to maximize the nodesâĂŹ contribution to the network. SORA uses a virtual markets approach where nodes sell resources as goods with associated prices that are set by the programmer. The network operates by nodes attempting to maximize their profit while staying within an energy budget. Thus, nodes are modeled as self-interested agents attempting to maximize their profit by performing local actions in response to global price information. This approach allows resource constrained nodes to run a simple cost-evaluation function that gives rise to sophisticated global behavior that can be controlled by adjusting advertised prices. Nodes adjust their behavior by learning the utility of the resources they can provide through payment feedback. This allows nodes to individually tune their schedules using reinforcement learning. Nodes receive virtual payment for actions they take. An action that is useful and contributes to the overall network goal is rewarded, while actions that do not benefit the required global behavior of the network are not. Thus, over time nodes learn which actions are profitable based on this feedback. A programmer can use the same mechanism to retask a network by simply adjusting the prices of resources, or actions. SORA controls the network lifetime by enforcing a local energy budget on node. The energy budget assumes that nodes are aware of the amount of energy actions consume. The authors model the energy budget using a token bucket where each node has a bucket of energy with a maximum capacity of C joules that is replenished at a specified rate. The rate represents the average desired energy usage rate. At each action taken by a node, the corresponding energy consumption is deducted from the bucket. If the available energy is less than the amount needed for an action, the node goes to sleep instead of performing it, thus conserving its battery. Using this simple approach, all a node has to do is monitor its local state and the global price vector and periodically select the action that maximizes its utility. The expected profit of actions varies over time due to price adjustments. In order to enable nodes to explore options instead of settling on an action that maximized profit at a particular time, SORA uses an ϵ-greedy action selection policy where the node selects the action that maximizes the expected profit with a probability $1 - \epsilon$ for a small value of ϵ. But with probability ϵ, the node selects an action at random from all of the available actions.

ADRA highlights the complications in distributed resource allocation and presents an approach where simple local actions performed by nodes gives rise to a desired global behavior. SORA presents an interesting approach to distributed resource allocation. While it is used to help nodes decide which actions to take at any point in time, a similar virtual market approach can be used to allow mobile devices to select T-HPCs to which tasks should be offloaded. For instance, the price of offloading to an HPC asset could be inversely proportional to its current load, or the quality of the network link to it, so that mobile devices do not overburden a single HPC machine or select a machine with an unstable connection. In the processing phase, nodes receive information on targets from their neighbors and fuse it with their own detected target information. It then computes the change in utility based on the information from its neighbors and computes a plan for its own sensor mode. Optionally, it can compute a plan for its neighbors' modes. It, then, sends the plan information to its neighbors. In the third phase, nodes receive plan information from their neighbors and resolve their plans with their neighbors' plans. Finally, it executes the plan to change its own sensor mode.

3.3 Offloading to Mobile Devices

Arslan et al. (*22*) present a distributed computing infrastructure, called Computing While Charging (CWC) that uses mobile devices to run tasks that are traditionally executed on servers. The authors motivate the solution as a way to efficiently use wasted compute cycles on mobile devices while they are charging overnight. The authors make four contributions: (i) profile the charging behavior of real phone owners to show the viability of CWC, (ii) allow programmers to execute parallelizable tasks on mobile devices easily, (iii) develop a simple task migration model to resume interrupted tasks, and (iv) implement and evaluate a prototype of CWC on 18 Android smartphones.

Shi et al. (*23*) describe Serendipity, which is a framework to enable the offloading of applications from smartphones to other mobile devices. The authors present a job model where the basic job component is a PNP-block composed of a *preprocess* program, *n* parallel *task* programs, and a *postprocess* program. The preprocess program processes the input data, such as splitting the input into multiple segments, and passes them to the parallel tasks. The postprocess program processes the output of the tasks, such as collecting and collating them.

Serendipity represents jobs graphically as directed acyclic graphs (DAGs) of PNP-blocks and is composed of a job engine, job profiler, job initiator, master, and a collection of workers. The job engine takes a script specifying the DAG, the programs, their execution profiles (e.g., CPU cycles) for all PNP-blocks, and the input data from the user. The authors do not describe how to

construct accurate execution profiles due to it being a challenging problem and out of the scope of the paper. The job profiler checks the script and constructs a complete job profile, which describes the execution time and energy consumption on every node, using the job execution profile and device profiles of the nodes. If the script checks out, the job engine launches a new job initiator, which store the job information in the local storage until the job completes. It is responsible for launching PNP-blocks when their parents have completed by running the preprocess program on a local worker and assigning a time-to-live (TTL), a priority, and a worker to every task. The TTL specifies the time before the result of a task should return. If a task misses its TTL, it is executed locally. The priority determines the relative importance of different tasks composing a job. A worker can be a single node or a set of nodes. Tasks are disseminated by the job engine, which is responsible for scheduling using information collected during the initial contact with other mobile devices. During this handshaking phase, devices exchange their profiles, residual energy, and a summary of tasks they have been assigned. The job engine can decide whether to execute a task locally, or disseminate it to another device in order to reduce the job completion time or conserve energy. The master receives a task from the job engine and starts a worker for it. It monitors the execution and returns the output to the job initator when the task finishes, using an underlying routing protocol such as Max-Prop. If an exception is thrown during execution, the master reports it to the job initator who terminates the job and reports the exception to the user. The authors assume all nodes are collaborative and trustworthy, but are aware that in certain applications, malicious nodes may exist. They state that a reputation-based trust protocol could be used to deal with such nodes.

The authors propose three different task allocation algorithms for Serendipity targeting three different scenarios in terms of contact predictability and the availability of a control channel. For the ideal network where contact between devices is predictable and there is a control channel the Water Filling algorithm is used to schedule tasks. The Water Filling algorithm first estimates the dissemination time for every task to every node. With this information and the estimated time to execute the tasks on every node, the algorithm can estimate the time at which the task will finish. With this time, the algorithm computes the time at which the output is sent back. Then, the algorithm picks, for each task, the node that achieves the minimum task completion time. It repeats the process for subsequent tasks taking the previous task schedule into account. The job initiator reserves the task execution time on all selected nodes and shares this information with other job initiators for future scheduling. For scenarios where contact is predictable but there is no control channel, the authors present a Computing on Dissemination with predictable contacts (pCoD) algorithm, since it is impossible to reserve task execution time in advance. Instead of explicitly assigning tasks to nodes, Computing on Dissemination (CoD) opportunistically

disseminated tasks to nodes that the mobile device encounters until the tasks finish executing. Based on the metadata exchanged when two phones meet, each phone decides which set of tasks to disseminate to the other phone. This decision is made in an attempt to minimize the task completion time of every task using the metadata from encountered nodes. CoD first estimates the execution time of its carried tasks on the other node based on the job profiles and device profile. For each task, it estimates the task completion time of executing locally as well as on the encountered device. If the local execution time is greater than the remote execution time, the task is assigned to the encountered node. For the third case, when contacts are unpredictable and there is no control channel, the authors present CoD with unpredictable contacts (upCoD), which constrains CoD with the lack of future contact information. In such a situation, the algorithm ignores data transfer time and attempts to minimize the execution time of the last task. While these algorithms schedule tasks in an attempt to minimize their execution time, they do not consider energy. In order to extend them to be energy aware, the authors present a utility function that can replace execution time in all three algorithms. The function attempts to consume less energy while avoiding nodes with low energy. Serendipity is tested both on an Emulab testbed as well as on Android phones.

CirrusCloud(*28*) extends the ideas of Serendipity, which targets offloading only to other mobile devices, to a generalized cyber-foraging platform. It is a work-in-progress of a framework to enable phones to offload to whatever resources they encounter—from central cloud computers to cloudlets to mobile devices. CirrusCloud extends CloneCloud, which is a system that automatically partitions mobile applications and offloads the computationally intensive aspects to the Cloud, to be robust to intermittant connectivity. The authors present an algorithm that chooses migration points of an application to minimize its execution time in the presence of intermittant connectivity. CirrusCloud is evaluated on data collected on the connectivity between a tablet that was carried on a bus and the WiFi access points on a campus. The authors show that CirrusCloud outperforms CloneCloud in this scenario when offloading to a central cloud and summarize Serendipity as the approach for offloading to mobile devices. The authors identify a number of issues that they are working on, including how to provide continuous execution if a mobile device loses contact with a cloudlet before it completes the processing of an allocated task, and how to optimally use the mixture of available resource types to maximize benefit.

CWC requires a centralized server to schedule tasks on phones, which will not be feasibile in a military scenario. Also, offloading in a tactical environment should occur when phones are being used, not when they are charging. Serendipity and CirrusCloud provides a scheduler that could be extended for offloading to HPC assets, yet leave a number of avenues open for further exploration and optimization. One avenue that is critical to be addressed is security. A

reputation-based security scheme may be sufficient for the general public, but is not adequate in a military environment. Such a setting requires a device to behave maliciously before it builds up a bad reputation. In a military system, a malicious or malfunctioning device cannot be tolerated as it could jeopardize a mission that relies on the mobile devices for computation. As Soldiers rely more and more on their mobile devices it will become even more critical to ensure the system is reliable, robust, and secure—a reputation-based system would not provide such guarantees.

4. Requirements

A system for battlefield computation on mobile devices would require the following characteristics:

- General and extensible
  - Provide an easy to use general purpose programming abstraction
  - Independent of underlying hardware infrastructure
- User friendly
  - Minimize burden on application developers
    - Support common programming languages
    - Provide a single-machine programming abstraction
  - Minimize burden on end-users
    - User interface that is easy to learn by a Soldier
    - Abstract away underlying hardware and networks
  - Maximize automation
    - Automatically classify and decompose programs
    - Automatically create and maintain network
    - Automatically learn network behavior and environment characteristics
- Intelligent
  - Identify changes in the communication environment, network topology, and computation needs

  - Tune algorithm parameters as topology and environment changes
  - Satisfy QoS requirements such as lifetime, deadlines, accuracy, etc.
  - If possible, modify the network topology to maximize performance and energy efficiency
- Secure and private
  - Protect from attacks such as jamming, Denial-of-Service (DoS), black hole, man-in-the-middle, etc.
  - Prevent the spread of malicious code such as viruses and worms
  - Minimize the information that can be inferred through snooping and packet sniffing

In order to implement the envisioned system, a number of subsystems have to be built and evaluated. These systems may already have been implemented in part and thus an attempt should be made to leverage the state of the art. The following are some of the systems necessary for computation on mobile devices in the battlefield:

1. **Ad-Hoc communication:** The battlefield computation network requires protocols to enable mobile devices to communicate without relying on cell towers or Wi-Fi access points. The network should be formed in an ad-hoc manner and be robust to changes in connectivity. The same network should allow communication with the mobile HPC assets. Possible approaches include multihop Bluetooth networks (*29–31*) and Wi-Fi direct. Mobile Ad-Hoc Network (MANET) routing protocols (*32–37*) and delay-tolerant networking (DTN) approaches (*38–43*) could be used to overcome the harsh wireless environment that can be expected in varying tactical environments.
2. **Network security:** The network should be secure to malicious attacks and faults. Wireless network security has been extensively studied and approaches for defense have been proposed (*44–48*). The system's vulnerabilities have to be identified and a suite of defense mechanisms developed and evaluated. The security solution for the network could be an integration of existing solutions, as long as all identified vulnerabilities are secured.
3. **Intelligent offloading:** As described above, a number of approaches to offloading computation from mobile devices have been proposed. The system requires an offloading approach that intelligently offload computation in an attempt to satisfy QoS requirements. For instance, maximizing the lifetime of all devices, maximizing the lifetime of the network, completing a computation before a deadline, etc. This requires a distributed

scheduling algorithm that enables a mobile device to decide if it should offload a task, and if so, to which device or set of devices.

4. **Intuitive user interface:** The user interface should be easy to use via the small screens of mobile devices for a Soldier who may not have extensive technical experience. A Soldier should be able to obtain a result to a computation in a quick and efficient manner while satisfying QoS requirements without compromising the rest of the network. For instance, a single Soldier should not inadvertantly consume a majority of the bandwidth or computation power, thus unwittingly launching a DoS attack on the network. At the same time, a Soldier should be able to easily make QoS tradeoffs such as system lifetime for computational speed.

5. **General purpose programming abstraction and execution framework:** The offloading and scheduling infrastructure of the system should be sufficiently expressive to support a large class of applications. The programming infrastructure should minimize the burden placed on programmers to annotate code for decomposition or offloading.

5. Army Relevance

Over the last couple of years there has been an increase in Soldiers using smartphones on the front lines. As more Army applications are developed, Soldiers will rely on their mobile devices more as an essential tool in their arsenal. In order for Soldiers to be able to use their devices for longer periods of time between recharges, offloading is being investigated as an option to maximize battery life. An essential part of an offloading solution in a tactical environment is intelligently selecting the resource to which an application is assigned.

6. Technical Challenges

The main technical challenges arise from the unpredictability of the system. Due to changes in devices, connectivity, and applications to be offloaded, no prior information can be assumed and system state has to be constantly updated. A scheduler for tactical computation offloading needs the following:

- An algorithm to efficiently update all smartphones with the global system state

- An algorithm to decide the optimal location to execute a particular application
- A method to offload computation and receive results efficiently in an environment with intermittent connectivity and high mobility
- An interface through which QoS constraints can be input
- A method for multihop communication over mobile devices
- A method for mobile devices to communicate with a tactical HPC asset
- An algorithm for efficient resource discovery

7. Conclusions

In order to enable battlefield computation on mobile devices a complete theory of distributed computation over device-heterogeneous ad-hoc networks in the battlefield is necessary. These devices can range from mobile devices to high-performance clusters. In addition, an algorithm to efficiently allocate powered computing resources for offloading computation is necessary. The algorithm should be able to satisfy QoS requirements, such as task execution deadlines, in an environment with unreliable communication links and high device mobility.

8. References

1. Manning, P. Army Stronger with Androids. Fox News, 2012.
2. Mlot, S. Army to Deploy Android Smartphones in Oct.. PC Magazine, 2012.
3. Beidel, E. Soldiers Skeptical of Smartphones in Combat. National Defense Magazine, 2012.
4. Messmer, E. U.S. Army Wants Soldiers to Have Advanced Smartphones, Wireless Technology. *Network World* **2011**, .
5. Ackerman, S. It Only Took the Army 16 Years and 2 Wars to Deploy This Network. Wired, 2012.
6. Sengupta, S. U.S. Military Hunts for Safe Smartphones for Soldiers. The New York Times, 2012.
7. Ackerman, S. Army Shows Off Soldier Smartphone Beta. Wired, 2011.
8. Kerr, D. DARPA Fortifies Soldiers' Smartphones Against Malware. *CNET* **2012**, .
9. McGarry, B. Army Set to Introduce Smartphones Into Combat. http://www.military.com/, 2013.
10. Horn, L. Army Might Give Troops Smartphones Soon. PC Magazine, 2011.
11. Montalbano, E. Army Expanding Soldier Smartphone Program. Information Week, 2010.
12. Kemp, R.; Palmer, N.; Kielmann, T.; Bal, H. Cuckoo: A Computation Offloading Framework for Smartphones. In *Mobile Computing, Applications, and Services*; Vol. 76; Gris, M., Yang, G., Eds.; Springer Berlin Heidelberg: 2012; pp 59–79.
13. Cuervo, E.; Balasubramanian, A.; Cho, D.-K.; Wolman, A.; Saroiu, S.; Chandra, R.; Bahl, P. MAUI: Making Smartphones Last Longer With Code Offload. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, ACM: New York, NY, USA, 2010.
14. Gordon, M. S.; Jamshidi, D. A.; Mahlke, S.; Mao, Z. M.; Chen, X. COMET: Code Offload by Migrating Execution Transparently. In *Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation*, USENIX Association: Berkeley, CA, USA, 2012.
15. Kosta, S.; Aucinas, A.; Hui, P.; Mortier, R.; Zhang, X. Thinkair: Dynamic Resource Allocation and Parallel Execution in the Cloud for Mobile Code Offloading. In *INFOCOM, 2012 Proceedings IEEE*, IEEE: Orlando, FL USA, March 2012.
16. Shires, D.; Henz, B.; Park, S.; Clarke, J. Cloudlet Seeding: Spatial Deployment for High Performance Tactical Clouds. In *PDPTA'12*, WORLDCOMP'12: Las Vegas, NV, USA, July 2012.

17. Briceño, L. D.; Siegel, H. J.; Maciejewski, A. A.; Oltikar, M.; Brateman, J.; White, J.; Martin, J.; Knapp, K. Heuristics for Robust Resource Allocation of Satellite Weather Data Processing on a Heterogeneous Parallel System. *Parallel and Distributed Systems, IEEE Transactions on* **2011**, *22* (11), 1780–1787.
18. Briceno, L. D.; Khemka, B.; Siegel, H. J.; Maciejewski, A. A.; Groër, C.; Koenig, G.; Okonski, G.; Poole, S. Time Utility Functions for Modeling and Evaluating Resource Allocations in a Heterogeneous Computing System. In *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*, Institute of Electrical and Electronics Engineers (IEEE): Anchorage, AK, USA., May 2011 Authorized distributor of all IEEE proceedings.
19. Gerkey, B. P.; Matarić, M. J. A formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research* **2004**, *23* (9), 939–954.
20. Matarić, M. J.; Sukhatme, G. S.; Østergaard, E. H. Multi-Robot Task Allocation in Uncertain Environments. *Autonomous Robots* **2003**, *14* (2), 255–263.
21. Mainland, G.; Parkes, D. C.; Welsh, M. Decentralized, Adaptive Resource Allocation for Sensor Networks. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation*; Vol. 2, USENIX Association: Berkeley, CA, USA, 2005.
22. Arslan, M. Y.; Singh, I.; Singh, S.; Madhyastha, H. V.; Sundaresan, K.; Krishnamurthy, S. V. Computing While Charging: Building a Distributed Computing Infrastructure Using Smartphones. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, ACM: New York, NY, USA, 2012.
23. Shi, C.; Lakafosis, V.; Ammar, M. H.; Zegura, E. W. Serendipity: Enabling Remote Computing Among Intermittently Connected Mobile Devices. In *Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing*, ACM: New York, NY, USA, 2012.
24. Young, B.; Apodaca, J.; Briceño, L.; Smith, J.; Pasricha, S.; Maciejewski, A.; Siegel, H. J.; Khemka, B.; Bahirat, S.; Ramirez, A.; Zou, Y. Deadline and Energy Constrained Dynamic Resource Allocation in a Heterogeneous Computing Environment. *The Journal of Supercomputing* **2013**, *63* (2), 326-347.
25. Kuhn, H. W. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly* **2006**, *2* (1-2), 83–97.
26. Cloud Software, openstack. http://www.openstack.org/, 2013.
27. Lim, H. B.; Lam, V. T.; Foo, M. C.; Zeng, Y. Adaptive Distributed Resource Allocation in Wireless Sensor Networks. In *Mobile Ad-hoc and Sensor Networks*; Vol. 4325, Springer Berlin Heidelberg: 2006; pp 770–781.

28. Shi, C.; Ammar, M. H.; Zegura, E. W.; Naik, M. Computing in Cirrus Clouds: The Challenge of Intermittent Connectivity. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ACM: New York, NY, USA, 2012.
29. Petrioli, C.; Basagni, S.; Chlamtac, M. Configuring BlueStars: Multihop Scatternet Formation for Bluetooth Networks. *Computers, IEEE Transactions on* **2003**, *52* (6), 779–790.
30. Li, X.; Wu, C.; Wang, X.; Gu, M.; Li, X.-Y.; Xuan, D. BlueSky: Realizing Buried Potential of Bluetooth to Sustain a Large-Scale Multi-Hop Network. *ArXiv e-prints* **2013**, ; Provided by the SAO/NASA Astrophysics Data System.
31. Basagni, S.; Petrioli, C. Multihop Scatternet Formation for Bluetooth Networks. In *Vehicular Technology Conference, 2002. VTC Spring 2002. IEEE 55th*; Vol. 1, IEEE: Birmingham, AL, May 2002.
32. Gorantala, K. "Routing Protocols in Mobile Ad-Hoc Networks", Master's thesis, Umea University, 2006.
33. Royer, E. M.; Toh, C.-K. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. *Personal Communications, IEEE* **1999**, *6* (2), 46–55.
34. Hong, X.; Xu, K.; Gerla, M. Scalable Routing Protocols for Mobile Ad-Hoc Networks. *Network, IEEE* **2002**, *16* (4), 11–21.
35. Singh, S.; Woo, M.; Raghavendra, C. S. Power-Aware Routing in Mobile Ad-Hoc Networks. In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, ACM: New York, NY, USA, 1998.
36. Akkaya, K.; Younis, M. A Survey on Routing Protocols for Wireless Sensor Networks. *Ad-Hoc Networks* **2005**, *3* (3), 325–349.
37. Taneja, S.; Kush, A. A Survey of Routing Protocols in Mobile Ad-Hoc Networks. *International Journal of Innovation, Management and Technology* **2010**, *1* (3), 2010–0248.
38. Musolesi, M.; Mascolo, C. Car: Context-Aware Adaptive Routing for Delay-Tolerant Mobile Networks. *Mobile Computing, IEEE Transactions on* **2009**, *8* (2), 246–260.
39. Mascolo, C.; Musolesi, M. SCAR: Context-Aware Adaptive Routing in Delay Tolerant Mobile Sensor Networks. In *Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing*, ACM: New York, NY, USA, 2006.
40. Musolesi, M.; Hailes, S.; Mascolo, C. Adaptive Routing For Intermittently Connected Mobile Ad-Hoc Networks. In *World of wireless mobile and multimedia networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, IEEE Computer Society: Washington, DC, USA, 2005.
41. Pelusi, L.; Passarella, A.; Conti, M. Opportunistic Networking: Data Forwarding In Disconnected Mobile Ad-Hoc Networks. *Communications Magazine, IEEE* **2006**, *44* (11), 134–141.

42. Wang, Y.; Wu, H. Delay/fault-tolerant Mobile Sensor Network (dft-msn): A New Paradigm For Pervasive Information Gathering. *Mobile Computing, IEEE Transactions on* **2007**, *6* (9), 1021–1034.
43. D'Souza, R.; Jose, J. Routing Approaches in Delay Tolerant Networks: A Survey. *International Journal of Computer Applications* **2010**, *1* (17), 8–14.
44. Kannhavong, B.; Nakayama, H.; Nemoto, Y.; Kato, N.; Jamalipour, A. A Survey Of Routing Attacks in Mobile Ad-Hoc Networks. *Wireless Communications, IEEE* **2007**, *14* (5), 85–91.
45. Karmore, P.; Bodkhe, S. A Survey on Intrusion in Ad-Hoc Networks and its Detection Measures. *International Journal on Computer Science and Engineering* **2011**, *3* (5), 1896–1903.
46. Jhaveri, R. H.; Patel, S. J.; Jinwala, D. C. D.O.S. Attacks in Mobile Ad-Hoc Networks: A Survey. In *Advanced Computing & Communication Technologies (ACCT), 2012 Second International Conference on*, Institute of Electrical and Electronics Engineers (IEEE): Rohtak, Haryana, India, January 2012.
47. Kaur, G.; Jain, V.; Chaba, Y. Wormhole Attacks: Performance Evaluation of On-Demand Routing Protocols in Mobile Ad-Hoc Networks. In *Information and Communication Technologies (WICT), 2011 World Congress on*, IEEE: Mumbai, India, December 2011.
48. Mamatha, G.; Sharma, D. S. A New Combination Approach to Secure MANETS Against Attacks. *International Journal of Wireless & Mobile Networks (IJWMN)* **2010**, *2*, 1–10.

List of Symbols, Abbreviations, and Acronyms

HPC	High Performance Computer
GPS	Global Positioning System
PC	Personal Computer
AI	Artificial Intelligence
QoS	Quality of Service
pmf	Probability mass function
SQ	Shortest Queue
MECT	Minimum Expected Completion Time
LL	Lightest Load
RAND	Random
HPT	High-Priority Task
RGT	Revenue Generation Task
EC2	Elastic Compute Cloud
AWS	Amazon Web Services
IaaS	Infrastructure as a Service
NASA	National Aeronautics and Space Administration
T-HPC	Tactical High Performance Computer
ADRA	Adaptive Distributed Resource Allocation
SORA	Self-Organizing Resource Allocation
CWC	Computing While Charging
DAG	Directed Acyclic Graph

TTL	Time-to-live
pCoD	Predictable contacts
CoD	Computing on Dissemination
upCoD	Unpredictable contacts
DoS	Denial-of-Service
MANET	Mobile Ad-Hoc Network
DTN	Delay-Tolerant Networking

NO. OF COPIES	ORGANIZATION
1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL IMAL HRA MAIL & RECORDS MGMT
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
5 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIH S TAMIM SOOKOOR DALE SHIRES DAVID BRUNO RONDA TAYLOR SONG PARK

INTENTIONALLY LEFT BLANK.